COMMONWEALTH OF AUSTRALIA

Copyright Regulations 1969

Warning

CHARLES DARWIN UNIVERSITY

Family Name	
Given Names	
Student Number	
Teaching Period	Semester 1, 2016

FINAL EXAMINATION	**DURATION**
HIT337 – Distributed Development	Reading Time: **10** minutes Writing Time: **180** minutes

INSTRUCTIONS TO CANDIDATES

A4 Sheet of Notes must be hand-written and submitted with final exam paper.

The examination has one section. All questions must be answered on the Answer Booklet provided. Please ensure that your name and student number are clearly indicated on your Answer Sheet and at the top of this examination paper.

- Note that questions ARE NOT of equal value.
- Read ALL questions carefully.
- Total marks – 40 marks.

EXAM CONDITIONS

You may begin writing from the commencement of the examination session. The reading time indicated above is provided as a guide only.

This is a RESTRICTED OPEN BOOK examination

No calculators are permitted

One A4 sheet of handwritten double-sided notes permitted

Hard copy, unannotated English translation dictionary only

ADDITIONAL AUTHORISED MATERIALS	EXAMINATION MATERIALS TO BE SUPPLIED
No additional printed material is permitted	1 x 16 Page Book

THIS EXAMINATION IS PRINTED DOUBLE-SIDED.

THIS PAGE HAS BEEN INTENTIONALLY LEFT BLANK.

HIT337 – Final Exam 2016

Question 1

a. Describe the 3 tier architecture. Describe what functionality goes in each tier. Remember the client (eg. the browser) is considered a separate fourth tier and should not be included.
(3 Marks)

b. Describe which java technologies are used to pass information between the tiers.
(2 Marks)

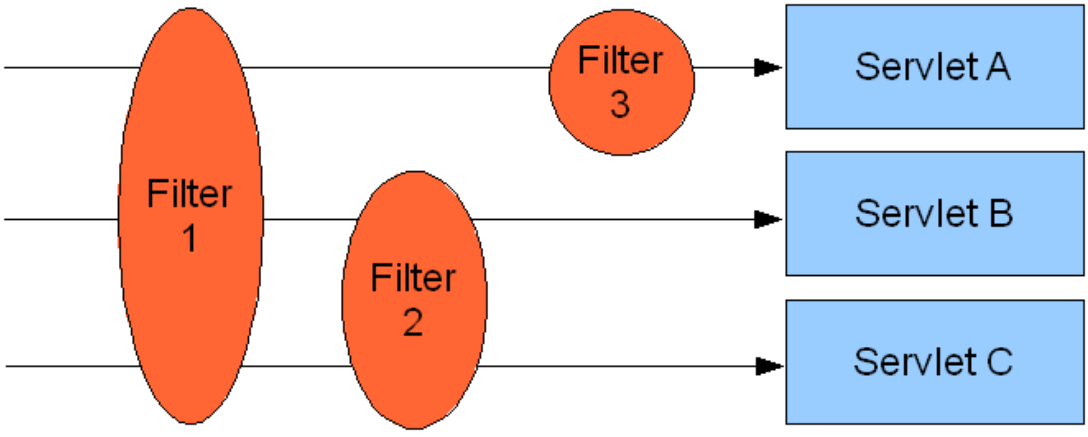
Question 2

Read the following excerpt from a web.xml document for the fictional web application found at the URL http://localhost/exam/

```
<web-app>
    <display-name>Exam webapp</display-name>
   <servlet>
       <servlet-name>question1</servlet-name>
       <servlet-class>exam.servlets.question1</servlet-class>
   </servlet>
   <servlet-mapping>
       <servlet-name>question1</servlet-name>
       <url-pattern>/question1</url-pattern>
   </servlet-mapping>
   <servlet>
       <servlet-name>question2</servlet-name>
       <servlet-class>exam.servlets.question2</servlet-class>
   </servlet>
   <servlet-mapping>
       <servlet-name>question2</servlet-name>
       <url-pattern>/question2</url-pattern>
   </servlet-mapping>
   <servlet>
       <servlet-name>question3</servlet-name>
       <servlet-class>exam.servlets.question3</servlet-class>
   </servlet>
   <servlet-mapping>
       <servlet-name>question3</servlet-name>
       <url-pattern>/question3</url-pattern>
   </servlet-mapping>
```

```
    <filter>
        <filter-name>checker1</filter-name>
        <filter-class>exam.filters.checker1</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>checker1</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
    <filter>
        <filter-name>checker2</filter-name>
        <filter-class>exam.filters.checker2</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>checker2</filter-name>
        <servlet-name>question2</servlet-name>
    </filter-mapping>
</web-app>
```

a. List all servlets and filters that appear in the web.xml (1 Mark)
b. List all known classes – use their full names (1 Mark)
c. Draw a diagram of the following requests and how they are parsed (similar to the following):



- http://localhost/exam/question1
- http://localhost/exam/question2
- http://localhost/exam/question3

(3 Marks)

Question 3

Describe the parts of an HTTP request. Describe the 2 main methods you can override and use in a class extending HttpServlet. How do they differ? Describe the Servlet lifecycle.

(5 Marks)

Question 4

Explain why you would put calls to a database using transaction methods with examples.

(4 Marks)

Question 5

Explain how JSP pages are compiled and their pros and cons when used.

(4 Marks)

Question 6

Take in two parameters, one called "key" and another called "message" e.g:
http://localhost/my_webapp/my_servlet?key=alpha&message=Hello_There

Provide a JSP page that will do the following:

The JSP should: Get the value of a **session** attribute with the same name as the value of the key request parameter. The value of this attribute will be an Integer. If this integer is greater than 0 then it should:

- Output the value of the message parameter in <h1> tags that many times. If the integer is less than or equal to 0 it should:
- Output the following message: "No message was output". **The JSP must satisfy the following:**

a. Contain the method: `Integer getSessionValue(HttpSession session, String key)` (which returns the integer obtained from the session.)

b. **Must not** write directly to the response using the out variable.

c. **Must** use scriptlets rather than JSTL tags.

d. The JSP page must produce a valid html page.

Question 6. (Continued)

Assumptions:

Don't bother checking if the request parameter and session attribute exist or not, just assume they will always be there.

Also assume the value in the session is an Integer, there is **no need** to do type checking.

(5 marks)

Question 7

Assuming that the following bean exists:

```
class Person {
  private String name;
  private int age;
  public Person(String name, int age) {
    this.name = name;
    this.age = age
  }
  public String getName() {
    return name;
  }
  public int getAge() {
return age; }
```

(Continued)

Question 7. (Continued)

And the following code has been run:

```
List<Person> people = new ArrayList();
Person a = new Person(“Michael”, 31)
people.add(a);
Person b = new Person(“John”, 43)
people.add(b);
etc..
request.setAttribute(“people”, people);
Map<Person, String> favouriteColours = new HashMap<Person,
String>();
favouriteColours.put(a, “red”);
favouriteColours.put(b, “blue”);
etc..
request.setAttribute(“colours”, favouriteColours);
```

Provide a tag file (not a jsp page) that accepts an attribute called “favouritePerson”

For each person it should output their details followed by a line break <br /> then execute the body of the tag making a variable available to the body of the tag with a name of “current_person”

For the details it should output their name, then age, then if they have a favourite colour it should be output it otherwise it should output “No favourite colour”.

When outputing the details it should output each person in the order they appear in the people array, except for the person with a name matching the “favouritePerson” attribute which should be output first.

For example. If the following URL was requested:

http://localhost/my_webapp/some_page.jsp?favouritePerson=John

It would output something like:

John 43 blue

Michael 31 red

`<body of tag>`

Jenny 45 No favourite colour

`<body of tag>`

`<body of tag>`

etc...

where `<body of tag>` is whatever is generated by the body of the tag.

This **must** be a tag file, not a tag handler.

You should use the JSTL tag library, (There should be **no** Java code or scriptlets).

You **must** use Expression Language rather than <jsp:useBeans> and <jsp:getProperty> tags

Provide a sample bit of JSP that uses this tag.

(6 marks)

Question 8

a) Provide a tag handler (Extend SimpleTagSupport) that takes three attributes i. an attribute called “titles” with a type of List<String>

ii. an attribute called “footer” with a type of JSPFragment iii. an attribute called “var” with a type of String

When run, the tag should perform the following: i. For each String in the “titles” attribute it should:

• Output the String in a header tag (eg. <h1>The string</h1>). • Run the body of the tag making a variable available to the body. The name of the

variable should be the value of the “var” attribute and the value of the variable

should be the current title String. ii. Finally at the end it should execute the “footer” fragment.

b) Provide the tag descriptor that you would put in a **TLD (Tag Library Descriptor)** for the above tag.

c) Provide a sample bit of JSP that uses this tag. The body of the tag should output the value of the variable that the tag makes available.

Assumptions:

• You can assume that there is an attribute attached to the request with a name of “page_titles” and a value of type List<String>. You should pass the value of this attribute to the “titles” attribute of the tag.

• Don’t worry about importing the tag just assume the tag has already been imported with a prefix of “myTags”

(6 Marks)